|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **결 과 보 고 서** | | | | |
| 학 과 | 학 년 | 학 번 | 조 | 성 명 |
| 전자공학과 | 3 | 12191505 | - | 윤수연 |
| 실험 제목 | Interrupt | | | |
| 1. **실험 과정**    1. 실습 1단계  |  | | --- | | #include<avr/io.h>  #include<util/delay.h>  #include <avr/interrupt.h>  unsigned char led\_on;  ISR(INT3\_vect){  if (led\_on) {  PORTB=0b10000000; // LED OFF  led\_on = 0;  }  else {  PORTB=0b00000000; // LED ON  led\_on = 1;  }  }  int main(void){  DDRB = 0b10001111; // 출력 설정  DDRD = 0b00000000; // 입력 설정  PORTD = 0b00001000; // 풀업저항 설정  EICRA = 0b10000000; //인터럽트 트리거 방식 설정  EIMSK = 0b00001000; //인터럽트 허용 설정  SREG |= 0x80; //전체 인트럽트 허가  led\_on = 1;  while(1) {  PORTB=0b00000111; // LED ON  \_delay\_ms(500);  PORTB=0b00000000; // 모든 LED OFF  \_delay\_ms(500);  }  } |   1단계 실험은 간단하게 interrupt를 사용하는 실험이다. 역시 ISR함수를 하나 구현한 것을 위의 사용한 코드에서 볼 수 있다. Shile문 안에는 기본적인 초기 setting만 있고 나머지 모든 주요 동작들은 ISR함수 안에서 찾아볼 수 있다. 각 핀과 레지스터 설정이 긴 편인데 모두 main문 안에 있다.   * 1. 실습 2단계     실습 2단계는 세 개의 버튼을 이용해 동작을 조작하는 실험이다. 각 버튼의 동작에 따라 ISR을 설정해주어 인터럽트를 대응시켜 준다. 실습 1단계에서는 인터럽트가 하나만 구현되었지만 여기서는 세 개다. 이 코드의 주요 알고리즘은 2주차의 코드와 같은데, 실제로 동작이 같기도 하다. 하지만 그 때는 여러 조건문과 함수를 이용했다면 ISR함수를 이용했다는 점이 가장 다르다. 역시 delay를 반복문 안에 일정 회수만큼 \_delay\_ms함수를 넣어 수를 소모하는 알고리즘을 적용했다.   1. **실험 결과**    1. 실습 1단계     그림 1. 실험 1 실행 직후    그림 2. 실험 1 버튼 누른 직후   * 1. 실습 2단계     그림 3. 실험 2 실행 모습 1    그림 4. 실험 2 실행 모습 2, 일시 중지    그림 5. 실험 2 실행 모습 3   1. **고찰**   이번주 수업은 이제까지 실험과 다르게 오프라인으로 진행했다. 각 소자들을 모두 처음 보는 것인데 너무 늦게 오프라인 수업에 참여하여 기본적인 setting을 하는 데에 있어서 어려움을 겼었다. 하지만 각 소자와 머리속의 proteus 프로그램의 소자가 매칭되면서 금방 해결할 수 있었다.  실험 1단계는 코드가 주어지기도 했고 간단한 알고리즘이지만 2단계 실험은 달랐다. 알고리즘 자체는 2주차 알고리즘과 같아서 아주 어렵지는 않았지만 하드웨어 특성 상 몇 번의 시행착오가 있었다. 가장 오랜 시간을 잡아먹었던 시행착오는 바로 딜레이에 관한 것이다. 크게 두 가지 고통을 받았는데, 처음은 delay를 처리하는 반복문을 내 노트북에서 돌릴 때와는 다르게 너무 빨리 처리해버려서 led가 계속 켜져있는 것처럼 보였다는 것이었다. Delay가 정말 작게 느껴졌다는 것이다. 하지만 이는 상식 내의 상황이고 이미 예상하기도 했기 때문에 바로 수정할 수 있었다.  두 번째 시행착오는 코드를 보면 main문에서, 그 중에서도 while문 안의 if문 안에 보면 state ==0b00000000이라는 조건이 있다. 이 때 state를 넣는 것이 알고리즘 특성 상 맞지만 실제로 돌려보니 저 조건을 인식하는 데에 있어서 너무 많은 시간이 걸렸다. 그래서 state대신에 그 자리를 대신할 수 있는 PORTB를 넣었더니 직접적인 PORT 언급이라 바로 인식하고 잘 수행하는 것을 볼 수 있었다. 하지만 알고리즘 상 state가 들어가는 것이 정확하기에 그렇게 실험을 수행했는데, if문의 역할에 따라 LED가 0번부터 7번까지 한 번씩 켜지는 주기 이후 다시 돌아가는 수행에서 긴 delay가 있다.  그 외에도 build 등 하드웨어로 실험하는 특성 상 여러 어려움이 있었지만 주어진 시간 내에 실험을 완수할 수 있었다. | | | | |
|  | | | | |